NATIONAL UNIVERSITY OF SINGAPORE

CS1010 – PROGRAMMING METHODOLOGY

(Semester 1: AY2014/15)

Time Allowed: 2 Hours

INSTRUCTIONS TO STUDENTS

1. This assessment paper consists of **THIRTEEN** questions and comprises **TEN** printed pages.
2. This is an **OPEN BOOK** assessment.
3. Calculators and electronic dictionaries are not allowed.
4. Answer all questions, and write your answers in the **ANSWER SHEETS** provided.
5. Fill in your Student Number with a pen, clearly on every odd-numbered page of your ANSWER SHEETS.
6. You may use **2B pencil** to write your programs.
7. Note the penalty will be given for programs that are unclear or unnecessarily long.
8. You must submit only the ANSWER SHEETS and no other document.

Section A: Multiple Choice Questions (MCQs) [12 marks]

There are six MCQs in this section. Each MCQ has one correct answer and is worth 2 marks. There is no penalty for wrong answer.

Q1. What is the final value of **x**?

```
int x = 0, i;
for (i = 1; i <= 4; i++) {
   switch ((i*3)%4) {
      case 0:
      case 1: x += 1; break;
      case 2: x += 2;
      case 3: x += 3;
   }
}
```

A. 6
B. 7
C. 9
D. 10
E. None of the above

Q2. What is the output of this code fragment?

```
char str1[] = "apple";
char str2[] = "pie";
strcpy(str1, str2);
printf("str = %s\n", strchr(str1,'p'));
```

A. pie
B. ple
C. pple
D. piele
E. pieapple

Q3. What is the result of **`f(7)`**?

```
int f(int n){
   if (n <= 3)
      return n;
   else
      return 3*f(n-1) - 2*f(n-2);
}
```

A. 9

B. 17

C. 33

D. 65

E. None of the above.

Q4. What is the final value of **`x`**?

```
char arr[5] = {'0', 1, '\0', '1', 0};
int i = 0, x = 0;

while (i++ < 4) {
   if (!arr[i] && x++) {
      ++x;
   }
}
```

A. 5

B. 4

C. 3

D. 2

E. 1

Q5. What does the expression **`rand()*2/RAND_MAX - 1`** generate?

Note: [*a*, *b*] means the range includes *a* and *b*; whereas (*a*, *b*) means the range excludes *a* and *b*.

A. A random real number in the range (-1, 1).

B. A random real number in the range [-1, 1].

C. A random integer in the range (-1, 1).

D. A random integer in the range [-1, 1].

E. None of the above.

Q6. What is the output of the following program?

```
typedef struct {
   char name[20];
   int age;
} student_t;

int main(void) {
   student_t s1 = {"alice", 20},
             s2 = {"bob", 18},
             s3 = {"carl", 21};
   strcpy(s1.name, s2.name);
   s2.age = s3.age;
   s3 = s1;
   printf("%s %d %s %d %s %d\n", s1.name, s1.age,
          s2.name, s2.age, s3.name, s3.age);
   return 0;
}
```

A. **alice 20 bob 21 alice 20**

B. **bob 20 bob 21 bob 20**

C. **bob 20 bob 20 bob 20**

D. **carl 21 bob 21 carl 21**

E. None of the above.

Section B: Short Structured Questions [14 marks]

Q7. Write the output of the following program. [2 marks]

```
#include <stdio.h>
#include <math.h>

int main(void){
   int a = 3, b = 1;
   int *p = &a, *q = &b;

   *q += *p;
   *p = (int) pow(*p, *q);

   printf("%d %d\n", *p, *q);

   return 0;
}
```

Q8. Write the output of the following code fragment. [4 marks]

```
int A[4][4] = {{2,0,1},{0,1},{1}};
int B[4] = {0};
int i, j;

for (i=0; i<4; i++)
   for (j=0; j<4; j++)
      A[i][j] += A[3-i][3-j];

for (i=0; i<4; i++)
   for (j=i; j<4; j++)
      B[i] += A[i][j];

printf("%d %d %d %d\n", B[0], B[1], B[2], B[3]);
```

Q9. Write the output of the following program. [8 marks]

The ASCII values for lower-case letters are given in the table below:

Letter	‘a’	‘b’	‘c’	‘d’	‘e’	‘f’	‘g’	‘h’	‘i’	‘j’	‘k’	‘l’	‘m’
ASCII value	97	98	99	100	101	102	103	104	105	106	107	108	109

Letter	‘n’	‘o’	‘p’	‘q’	‘r’	‘s’	‘t’	‘u’	‘v’	‘w’	‘x’	‘y’	‘z’
ASCII value	110	111	112	113	114	115	116	117	118	119	120	121	122

```
#include <stdio.h>

void f(char *, char);

int main(void) {
    char ch1 = 'a', ch2 = 'z';
    while (ch2 – ch1 > 3) {
        f(&ch1, ch2);
        printf("%c %c\n", ch1, ch2);

        f(&ch2, ch1);
        printf("%c %c\n", ch1, ch2);
    }
    return 0;
}

void f(char *a, char b) {
    *a = (*a + b) / 2;
    b = (*a + b) / 2;
}
```

Section C: Problem Solving [54 marks]

Q10. Calculating SAP [24 marks]

The following structure contains information about a student's results for a module.

```
typedef struct {
   char code[9];          // Module code, eg: "CS1010"
   int mc;                // Number of modular credits
   char grade[3];         // Grade
   char su[6];            // To S/U or not?
} module_t;
```

Grades are strings such as "A+", "A", …, "F". Member **su** is a string that contains either "TRUE" or "FALSE".

Study the **main()** function below:

```
int main(void) {
   module_t modules[7];  // A student may take up to 7 modules
   int num_modules;      // Actual number of modules he takes
   float sap;            // Semester Aggregate Point

   num_modules = readResults(modules);
   sap = computeSAP(modules, num_modules);
   printf("SAP = %.2f\n", sap);

   return 0;
}
```

(a) Write a function **readResults()** to read data from a text file called **results.in** into the **modules** array and return the number of modules read. The first line of the file contains the number of modules – at least 1 and at most 7 – the student took. This is followed by information about each module he took.

You are to fill in the parameter(s) of the function.

Note:

- You should check that the file can be opened properly and write appropriate code if the opening of file fails.
- The words "TRUE" and "FALSE" are spelled in mixed-case in the text file. You are to ensure that the **su** members in the **modules** array contain only uppercase letters.

An example of **results.in** is shown on the right. The student took 3 modules, the first is CS1010 which has 4 MCs. He obtained grade A+ for it, and did not S/U it.

```
3
CS1010
4
A+
False
MA1301
4
C
trUE
GEM1234
2
B-
fALse
```

[8 marks]

Q10. *(continue…)*

(b) Write a function

```
float gradeToPt(char grade[])
```

that takes in a grade (eg: "B+") and returns its corresponding point (eg: 4.0). The grades and their corresponding points are shown in the table below.

Letter Grade	A+	A	A-	B+	B	B-	C+	C	D+	D	F
Grade Point	5.0	5.0	4.5	4.0	3.5	3.0	2.5	2.0	1.5	1.0	0.0

Your function should declare and make use of two arrays: the **grades** array that contains the above grades, and the **pts** array that contains the above points. [8 marks]

(c) Write a function **`computeSAP()`** to compute the Semester Aggregate Point (SAP) the student obtained. If MC_i is the number of modular credits of module *i*, and GP_i the grade point the student obtained for module *i*, then the SAP is computed as follows:

$$\text{SAP} = \Sigma(\text{MC}_i \times \text{GP}_i)/\Sigma\text{MC}_i$$

The above formula is applied only to modules that are <u>not</u> S/U. In our example, CS1010 (4 MCs, A+ grade) and GEM1234 (2 MCs, B- grade) are not S/U, so

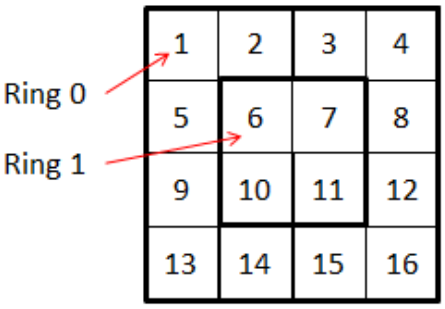
SAP = (4 × 5.0 + 2 × 3.0) / (4 + 2) = 26/6 = 4.33

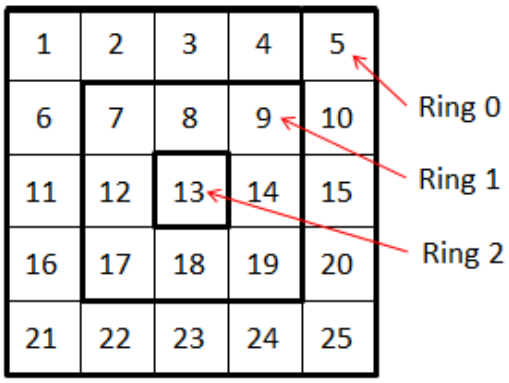
Your **`computeSAP()`** function should use the function **`gradeToPt()`** in part (b). Your function should return 0.0 if all the modules are S/U.

You are to fill in the parameter(s) of the function. [8 marks]

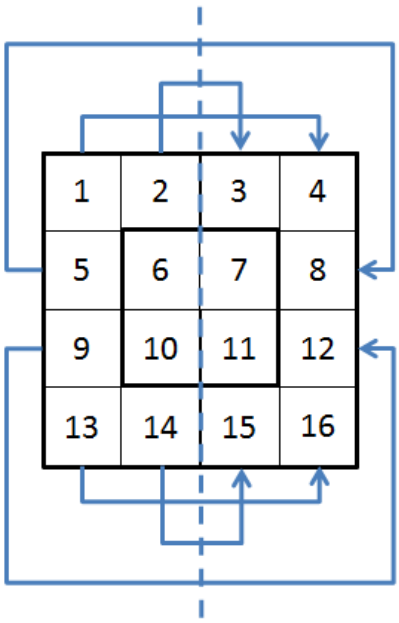
Q11. Squarelotron

A Squarelotron is a square matrix of numbers divided into rings which are indexed from outermost to inner most. For example, a Squarelotron of size 4 is as shown below on the left and it is divided into two rings: Ring 0 consists of 1-5, 8, 9 and 12-16, while Ring 1 consists of 6-7 and 10-11. Similarly, a Squarelotron of size 5 is as shown below on the right and divided into three rings.





Various operations can be performed on a ring in a Squarelotron. One operation is to flip from left to right a particular ring of the Squarelotron. For example, flipping Ring 0 of the first sample Squarelotron from left to right gives the Squarelotron below on the right.



4	3	2	1
8	6	7	5
12	10	11	9
16	15	14	13

Q11. *(continued…)*

Write a function

```
void flipLeftRight(int square[][10], int size, int index)
```

that flips ring **`index`** of the Squarelotron **`square`** from left to right. You may assume that the Squarelotron has at most 10 rows, and **`size`** is the actual number of rows (at most 10). [10 marks]

Q12. Period of a String

Given a string, find the length of the shortest substring such that the string can be formed by the concatenation of one or more occurrences of the substring. This length is also known as the **period** of the string.

For example, given the string "abcabcabcabc", it can be formed by 4 occurrences of the substring "abc" (of length 3), or 2 occurrences of the substring "abcabc" (of length 6) or 1 occurrence of the substring "abcabcabcabc" (of length 12). In this case, the period is 3.

As another example, given the string "abcab", it can only be formed by 1 occurrence of the substring "abcab", and hence the period is 5.

Write a function

```
int period(char str[])
```

to compute the **period** of the string **`str`**. [10 marks]

Q13. Note that parts (a) and (b) are independent. Hence, you may still do part (b) if you are unable to do part (a).

(a) Write a recursive function

`reverse(char str[], int start, int end)`

to reverse the characters in the string **`str`** from index `start` through `end`. For example, if **`str`** is "Programming", then calling **reverse(str, 3, 7)** will change **`str`** to "Prommarging".

No mark will be given if the function is not recursive. [5 marks]

(b) You are given a string that contains two substrings A and B one after the other, i.e. A followed by B. You are also given the lengths of A and B.

Write a function

`swapSubstrings(char str[], int len_a, int len_b)`

that takes in the string and the two lengths as parameters, and uses only the **reverse()** function in (a) and no other operation to swap the two substrings such that substring A comes after substring B. You may invoke **reverse()** more than once.

The function **swapSubstrings()** does not require recursion. You may assume that the sum of the lengths of A and B is the length of the given string.

For example, if **`str`** is "**`Hamburger`**" and the lengths of the substrings are 3 and 6, i.e. A is "**`Ham`**" and B is "**`burger`**", the resultant string after calling **swapSubstrings(str, 3, 6)** should be "**`burgerHam`**". [5 marks]

=== END OF PAPER ===